Sensors Service in WRT

Note that the Platform Services extensions were introduced for S60 5th edition, thus they do not work with older devices.

As with other Service APIs the interface for Sensors is retrieved using the device entry point. The service object is Service.Sensor and the interface is ISensor. Thus to retrieve the interface for the API you need to call following code in your application:

```
so = device.getServiceObject("Service.Sensor","ISensor");
```

Functions provided by the API

1. FindSensorChannel
2. RegisterForNotification
3. Cancel
4. GetChannelProperty
5. SetChannelProperty

Finding sensors

As different devices do provide different sets of sensors, you need to find out which sensors the device you are running on has. To do this, you could define the criteria to find all possible sensors as follows:

```
var SensorParams = {
SearchCriterion : "All"
};
var result = so.ISensor.FindSensorChannel(SensorParams);
```

The SearchCriterion could also be defined to find some specific sensor, with 5th edition following sensor identifiers has been defined:

- All
- AccelerometerAxis
- AccelerometerDoubleTapping
- Orientation
- Rotation

The result.ErrorCode provides error codes where the 0 means no error, the result.ErrorMessage would have the human readable error description.

The result.ReturnValue would then include the list of found channel maps, which you could iterate through to find out the sensors that are available in that particular device.

The map includes following 32 bit integer values:

- ChannelId: Unique ID representing the channel.
- ContextType: Defines the context in which the channel is available.
- Quantity: Defines the quantity being sensed.
- ChannelType: Defines a unique type id for each channel.
- DataItemSize: Data item size delivered in the channel.
- ChannelDataTypeId: Unique data type identifier for the data being sensed.

And following string values:

- Location: Location of the sensor related to channel.
- VendorId: Vendor Id of the sensor related to channel.

The ContextType values are defined as follows:

0: Not defined

1: Ambient sensor eg. To measure pressure

2: Gives info on device itself

3: Measures user initiated stimulus.

And the Quantity values are defined as follows:

0: Not defined

10: Acceleration

11: Tapping

(C) Copyright Nokia 2013. All rights reserved.

12: Orientation

13: Rotation

14: Magnetic

15: Tilt

Starting Sensor data retrieval

To start data retrieval, you need to first find the sensor so you can utilize the map for the sensor to start the data retrieval. You also need to define callback function that is called when there is data available for the sensor.

You can also have multiple sensors active at any given time. To start them all, you could use the full map list gotten from previous step, and just loop the result and start data retrieval for each sensor separately with following code:

```
for (var i = 0; i < result.ReturnValue.length; i++)
{
var SensorParams = {
    ListeningType: "ChannelData",
    ChannelInfoMap: result.ReturnValue[i]
};

var result2 = so.ISensor.RegisterForNotification ( SensorParams, SenssCallback);
}
```

You could also define the ListeningType as “ChannelPropertyChange”, which would then mean that you are not listening the actual data, but the changes on sensor properties.

Retrieving the sensor Data

After calling the RegisterForNotification, the callback defined will be called when ever there is data available for the requested sensor. The callback function could be defined for example as follows:

```
function SenssCallback(id1, id2, result){

 var returnvalue = result.ReturnValue;
 var datatype = returnvalue.DataType;
 if (datatype == "AxisData"){
  var xaxisdata = returnvalue.XAxisData;
  var yaxisdata = returnvalue.YAxisData;
  var zaxisdata = returnvalue.ZAxisData;
  // do something with the data.
 }
}
```

The result variable has the results for the selected sensor, the id1 identifies the transaction id that could be used for canceling original request, so no more dat would be supplied for the request.

With all results there is ReturnValue.TimeStamp that tells the time when the data was obtained, there is also always a datatype string identified ReturnValue.DataType, which can be used to determine the datatype for the sensor data, following types are defined:

- “AxisData” for AccelerometerAxis sensor
- “DoubleTappingData” for AccelerometerDoubleTapping sensor
- “OrientationData” for Orientation sensor
- “RotationData” for Rotation sensor
- “ChannelPropertyChange” for ChannelPropertyChange events.

Type specific additional data for AxisData for which all are 32 bit integer values include:

- XAxisData
- YAxisData
- ZAxisData

Type specific additional data for DoubleTappingData includes only one 32 bit integer value:

- DeviceDirection.

Type specific additional data for OrientationData includes only one string value: DeviceOrientation. This string can have following values:

- Undefined
- DisplayUp
- DisplayDown
- DisplayLeftUp
- DisplayRightUp
- DisplayUpwards
- DisplayDownwards

Type specific additional data for RotationData for which all are 32 bit integer values include:

- XRotation
- YRotation
- ZRotation

(C) Copyright Nokia 2013. All rights reserved.

Type specific additional data for ChannelPropertyChange includes following 32 bit integer values:

- ItemIndex
- PropertyDataType

, and following values that can either be 32 bit integers, doubles or strings:

- PropertyMinValue
- PropertyMaxValue
- PropertyValue

, and following Strings:

- DataType
- PropertyId

, as well as boolen value called: ReadOnly, that identified whether the property is read only or not.

The datatype defines the types for the property values, this identified can have following values:

0: For Integer datatype

1: For Double datatype

2: For String datatype

The PropertyId can have following values:

- DataRate
- Availability
- MeasureRange
- ChannelDataFormat
- ChannelAccuracy
- ChannelScale
- ScaledRange
- ChannelUnit
- SensorModel
- ConnectionType
- Description

Simple Example

Hello Sensor ZIP archive

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.